REMARKS

Upon entry of the foregoing amendments, claims 1-15 remain pending in the present application. Claims 1-10 have been amended. Claims 11-15 have been added. The subject matter of claims 1-15 can be found in the original application at least with respect to FIGs. 1-4 and the corresponding detailed description. Accordingly, Applicant respectfully submits that no new matter has been introduced to the application.

In response to item 3 (page 2) of paper no. 4, Applicant respectfully traverses the rejection of claims 2, 3, and 5 under 35 U.S.C. §112, second paragraph.

In response to item 5 (page 3)of paper no. 4, Applicant respectfully traverses the rejection of claims 1-10 under 35 U.S.C. §103(a) over *Hooker* in view of *Blasciak*.

Applicant respectfully submits that pending claims 1-15 are patentable over the cited art of record. Accordingly, reconsideration and allowance of the application and presently pending claims 1-15 are respectfully requested.

I. Claim Rejections Under 35 U.S.C. §112 - Claims 2, 3, and 5

A. Statement of the Rejection

The Office Action indicates that claims 2, 3, and 5 stand rejected under 35 U.S.C. 112, second paragraph as allegedly being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

B. Discussion of the Rejection

Applicant respectfully traverses the rejection of claims 2, 3, and 5. Applicant has amended independent claim 1 and dependent claims 2 and 3 that depend from claim 1. In addition, Applicant has amended independent claim 4 and dependent claim 5 that depends from claim 4. Applicant's amended claims 2, 3, and 5 include sufficient antecedent basis for claimed limitations. Accordingly, Applicant respectfully submits that the rejection of claims 2, 3, and 5 should be withdrawn.

II. Claim Rejections Under 35 U.S.C. §103(a) - Claims 1-10

A. Statement of the Rejection

The Office Action indicates that claims 1-10 stand rejected under 35 U.S.C. 103(a) as allegedly being unpatentable over U.S. Patent No.: 5,787,286, hereafter *Hooker*, in view of U.S. Patent No. 5,265,254, hereafter *Blasciak*.

B. Discussion of the Rejection

Applicant respectfully traverses the rejection of claims 1-10. In order for a claim to be properly rejected under 35 U.S.C. §103, "[t]he PTO has the burden under section 103 to establish a *prima facie* case of obviousness. To establish a *prima facie* case of obviousness, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).

In this regard, the cited references (*i.e.*, *Hooker* and *Blasciak*) do not show the combination of elements recited in Applicant's claimed invention. Thus, the cited references fail to meet the burden of disclosing, teaching, or suggesting each feature of Applicant's claimed invention. Consequently, for at least this reason the rejection fails to establish a *prima facie* case of obviousness when applied to Applicant's claims 1-10. Accordingly, for at least this reason, the claim rejections under 35 U.S.C. §103 should be withdrawn.

Specifically, and with particular regard to the claims, each of Applicant's amended independent claims include at least one element that is not disclosed, taught, or suggested by the apparatus for performing software performance checks apparently disclosed in *Hooker* and the system for debugging software through the use of code markers apparently disclosed in *Blasciak*.

In addition, Applicant disagrees with the assertion that one of ordinary skill in the art would be motivated to combine *Hooker's* tabulation instructions with the teachings of *Blasciak*. As explained in the Background of the Invention section, *Hooker* is directed to a real-time software solution for obtaining performance data that does not occupy processor time. See *Hooker*, column 1, lines 51-56. As further explained in the Disclosure of the Invention section, *Hooker* apparently discloses

inserting at least one tabulation instruction in a first plurality of instructions and executing the tabulation instruction using a second execution unit. By executing the tabulation instruction in a second execution unit, the system of *Hooker* can be used to monitor or otherwise measure the execution performance of the first plurality of instructions without degrading the execution performance of the first plurality of instructions. See *Hooker*, column 2, lines 5-24. As admitted by the Office in the statement of the rejection, *Hooker* does not specify the nature of, nor does *Hooker* suggest, inserting a correctness check function associated with a particular portion of the first set of instructions.

To overcome the admitted failure of *Hooker* to disclose, teach, or suggest Applicant's claimed invention, the statement of the rejection alleges that *Blasciak* teaches identifying insertion points where links to instructions can be applied to verify correctness of time-based margins, references, context switches, and branch conditions. Office Action, page 4, lines 9-17. The statement of the rejection then alleges that in view of the common intent by both *Hooker* and *Blasciak*, it would have been obvious for one of ordinary skill in the art at the time of the invention was made to implement the code insertion technique as taught by *Hooker* to add therein instrumentation code for correctness checking as suggested by *Blasciak* because a systematic and timely elimination of faulty references or memory inconsistencies, would enhance dynamic optimization of code execution, improve performance, and minimize additional resource usage.

Applicant respectfully disagrees with the assertion that the proposed combination of *Hooker* and *Blasciak* obviates Applicant's claimed method and apparatus. First, Applicant's claims, as amended, distinguish Applicant's claimed apparatus and method from the systems of both *Hooker* and *Blasciak*. Second, Applicant notes that *Hooker* and *Blasciak* do not share a common intent. *Hooker's* motivation is to monitor run performance without degrading execution performance of the first plurality of instructions. *Blasciak's* intent is to verify correctness of time-based margins, variable references, context switches, and branch conditions. In contrast with the intent of *Hooker* to not degrade execution performance, *Blasciak*, on its face, indicates that the method for identifying insertion points, inserting code markers, and linking instructions to the code marker, results in an intrusion into the underlying software. See *Blasciak*, column 3, lines 10-21. Consequently, one of

ordinary skill in the art would not be motivated to combine an intrusive code marker with links to additional code as apparently taught by *Blasciak* with *Hooker's* tabulation instructions, as the result of the proposed combination would negate the alleged advantages of the teachings of *Hooker*.

In this regard, the Office Action reiterates on page 7 of the Office Action that the common desirability of the cited references is to allow maximum code instrumenting and performance checking with minimized intrusion time. The Office Action misapplies this goal with respect to *Hooker* to justify the combination with *Blasciak*. As stated above, *Hooker* is apparently directed to accurate run time performance monitoring. *Hooker* specifically inserts tabulation instructions so that they can be bypassed entirely by a first execution unit. As a result, the run time performance of a first set of instructions as processed by a first execution unit is not degraded. Thus, *Hooker's* goal is in direct contradiction with the Office's assertion that *Hooker's* intention is to maximize code instrumenting and performance checking with minimized intrusion. Any intrusion beyond a tabulation instruction adversely degrades the execution run time of an instruction set. *Hooker* by all indications is unwilling to degrade execution performance. *Blasciak*, on the other hand, expressly teaches and explains the trade-off dilemma and is obviously willing to take the suggested execution degradation for the additional feedback provided by the inserted links to instrumentation code. For at least the reason that the Office mischaracterizes *Hooker*, Applicant submits that one skilled in the art would not be motivated to combine the systems and methods of *Hooker* with those apparently described in the *Blasciak* reference.

As stated above, Applicant's claimed apparatus and methods as recited in independent claims 1, 4, 6, and 9 are not rendered obvious over *Hooker* in view of *Blasciak*.

1. Claims 1-3

Applicant's independent claim 1 is exemplary. For convenience of analysis, independent claim 1, as amended, is repeated on the following page in its entirety.

> 1. An apparatus for performing correctness checks, the apparatus comprising:
>
> logic configured to receive a first set of instructions;
>
> ***logic configured to generate an initial instruction schedule and a conditional instruction stream from the first set of instructions***; such that the initial instruction schedule is devoid of code sequences comprising correctness check functions and such that code sequences of the conditional instruction stream are associated with a corresponding set of one or more instructions in the initial instruction schedule;
>
> ***logic configured to evaluate the initial instruction schedule*** to determine whether the initial instruction schedule includes spare instruction slots into which said code sequences associated with correctness check functions can be inserted into the initial instruction schedule ***such that a final instruction schedule responsive to the initial instruction schedule would not require a longer run time than the initial instruction schedule***; and
>
> ***logic configured to generate the final instruction schedule responsive to the initial instruction schedule, the conditional instruction stream, and the logic configured to evaluate.***

(Applicant's independent Claim 1 - *emphasis added.*)

The cited art of record fails to disclose, teach, or suggest at least the emphasized elements of pending claim 1 as shown above. Consequently, claim 1 is allowable.

Specifically, the systems disclosed in *Hooker* and *Blasciak* fail to disclose, teach, or suggest Applicant's claimed "***logic configured to generate an initial instruction schedule and a conditional instruction stream from the first set of instructions . . .***" Both *Hooker* and *Blasciak* are silent regarding logic configured to generate an initial schedule and a conditional instruction stream from the first set of instructions.

Furthermore, the systems disclosed in *Hooker* and *Blasciak* fail to disclose, teach, or suggest Applicant's claimed "***logic configured to evaluate the initial instruction schedule . . . such that a final instruction schedule responsive to the initial instruction schedule would not require a longer run time than the initial instruction schedule***." In this regard, both *Hooker* and *Blasciak* are silent regarding logic configured to evaluate an initial instruction schedule such that a final instruction schedule would not require a longer run time than the initial instruction schedule.

Hooker, as described above, merely teaches inserting tabulation instructions in a first instruction set. *Hooker* executes the instructions in a first execution unit and processes the tabulation instructions via a second execution unit. Inserting and processing tabulation instructions, as apparently taught by *Hooker*, does not disclose, teach, or suggest Applicant's claimed logic that evaluates a first instruction set. *Blasciak*, apparently teaches the insertion of code markers and linking instructions to the markers that can be used to verify correctness of time-based margins, variable references, context switches, and branch conditions. *Blasciak* cannot disclose, teach, or suggest Applicant's claimed logic configured to evaluate the initial instruction schedule for at least the reason that *Blasciak* results in an intrusion into the underlying software, whereas Applicant's claimed logic evaluates the initial instruction schedule such that a final instruction schedule responsive to the initial instruction schedule would not require a longer run time than the initial instruction schedule.

Moreover, because the systems disclosed in *Hooker* and *Blasciak* fail to disclose, teach, or suggest Applicant's claimed logic that evaluates a first instruction set, both *Hooker* and *Blasciak* cannot disclose, teach, or suggest Applicant's claimed "***logic configured to generate the final instruction schedule responsive to the initial instruction schedule, the conditional instruction stream, and the logic configured to evaluate***. Thus, claim 1 is allowable over the proposed combination and the rejection should be withdrawn.

Because independent claim 1 is allowable, dependent claims 2 and 3 are also allowable. *See In re Fine*, 837, F.2d 1071, 5 U.S.P.Q.2d 1596, 1598. (Fed. Cir. 1988.) Accordingly, Applicant respectfully requests that the rejection of claims 1-3 be withdrawn.

2. Claims 4 and 5

For convenience of analysis, independent claim 4, as amended, is repeated on the following page in its entirety.

> 4. An apparatus for performing correctness checks, the apparatus comprising:
>
> means for receiving a first set of instructions;
>
> ***means for generating an initial instruction schedule and a conditional instruction stream from the first set of instructions***, such that the initial instruction schedule is devoid of code sequences comprising correctness check functions and such that code sequences of the conditional instruction stream are associated with a corresponding set of one or more instructions in the initial instruction schedule;
>
> ***means for evaluating the initial instruction schedule*** to determine whether the initial instruction schedule includes spare instruction slots into which said code sequences associated with correctness check functions can be inserted into the initial instruction schedule ***such that a final instruction schedule responsive to the initial instruction schedule would not require a longer run time than the initial instruction schedule***; and
>
> means for inserting said code sequences associated with the correctness check function into the spare instruction slots if enough spare instruction slots exist in the initial instruction schedule for accommodating said code sequences.

(Applicant's independent Claim 4 - *emphasis added.*)

The cited art of record fails to disclose, teach, or suggest at least the emphasized elements of pending claim 4 as shown above. Consequently, claim 4 is allowable.

Specifically, the systems disclosed in *Hooker* and *Blasciak* fail to disclose, teach, or suggest Applicant's claimed "***means for generating an initial instruction schedule and a conditional instruction stream from the first set of instructions . . .***" Both *Hooker* and *Blasciak* are silent regarding means for generating an initial schedule and a conditional instruction stream from the first set of instructions.

Furthermore, the systems disclosed in *Hooker* and *Blasciak* fail to disclose, teach, or suggest Applicant's claimed "***means for evaluating the initial instruction schedule . . . such that a final instruction schedule responsive to the initial instruction schedule would not require a longer run time than the initial instruction schedule.***" In this regard, both *Hooker* and *Blasciak* are silent regarding evaluating an initial instruction schedule such that a final instruction schedule would not require a longer run time than the initial instruction schedule. *Hooker*, as described above, merely teaches inserting tabulation instructions in a first instruction set. *Hooker*

executes the instructions in a first execution unit and processes the tabulation instructions via a second execution unit. Inserting and processing tabulation instructions, as apparently taught by *Hooker*, does not disclose, teach, or suggest Applicant's claimed means for evaluating a first instruction set. *Blasciak*, apparently teaches the insertion of code markers and linking instructions to the markers that can be used to verify correctness of time-based margins, variable references, context switches, and branch conditions. *Blasciak* cannot disclose, teach, or suggest Applicant's claimed means for evaluating the initial instruction schedule for at least the reason that *Blasciak* results in an intrusion into the underlying software (*i.e.*, increased execution time), whereas Applicant's claimed means for evaluating the initial instruction schedule determines whether the insertion of a code sequence from the conditional instruction stream would require a longer run time than the initial instruction schedule. If so, the code sequence is discarded or otherwise ignored when constructing the final instruction schedule. For at least these reasons, claim 4 is allowable over the proposed combination and the rejection should be withdrawn.

Because independent claim 4 is allowable, dependent claim 5 is also allowable. *See In re Fine*, *supra*. Accordingly, Applicant respectfully requests that the rejection of claims 4 and 5 be withdrawn.

3. Claims 6-8

For convenience of analysis, independent claim 6, as amended, is repeated below in its entirety.

> 6. A method for performing correctness checks, the method comprising the steps of:
>
> receiving a first set of instructions;
>
> ***generating an initial instruction schedule and a conditional instruction stream from the first set of instructions***, such that the initial instruction schedule is devoid of code sequences comprising correctness check functions and such that code sequences of the conditional instruction stream are associated with a corresponding set of one or more instructions in the initial instruction schedule;
>
> ***evaluating the initial instruction schedule*** to determine whether the initial instruction schedule includes spare instruction slots into which said code sequences associated with correctness check functions can be inserted into the initial instruction schedule ***such that a final***

> ***instruction schedule would not require a longer run time than the initial instruction schedule***; and
>
> inserting said code sequences from the conditional instruction stream and associated with the correctness check function into the spare instruction slots if enough spare instruction slots exist in the initial instruction schedule for accommodating said code sequences.

(Applicant's independent Claim 6 - *emphasis added.*)

The cited art of record fails to disclose, teach, or suggest at least the emphasized elements of pending claim 6 as shown above. Consequently, claim 6 is allowable.

Specifically, the systems disclosed in *Hooker* and *Blasciak* fail to disclose, teach, or suggest Applicant's claimed limitation of "***generating an initial instruction schedule and a conditional instruction stream from the first set of instructions . . .***" Both *Hooker* and *Blasciak* are silent regarding generating an initial schedule and a conditional instruction stream from the first set of instructions.

Furthermore, the systems disclosed in *Hooker* and *Blasciak* fail to disclose, teach, or suggest Applicant's claimed limitation of "***evaluating the initial instruction schedule . . . such that a final instruction schedule would not require a longer run time than the initial instruction schedule***." In this regard, both *Hooker* and *Blasciak* are silent regarding evaluating an initial instruction schedule such that a final instruction schedule would not require a longer run time than the initial instruction schedule. *Hooker*, as described above, merely teaches inserting tabulation instructions in a first instruction set. *Hooker* executes the instructions in a first execution unit and processes the tabulation instructions via a second execution unit. Inserting and processing tabulation instructions, as apparently taught by *Hooker*, does not disclose, teach, or suggest Applicant's claimed evaluating a first instruction set. *Blasciak*, apparently teaches the insertion of code markers and linking instructions to the markers that can be used to verify correctness of time-based margins, variable references, context switches, and branch conditions. *Blasciak* cannot disclose, teach, or suggest Applicant's claimed limitation of evaluating the initial instruction schedule for at least the reason that *Blasciak* results in an intrusion into the underlying software (*i.e.*, increased execution time), whereas Applicant's claimed evaluating the initial instruction schedule determines whether a final instruction schedule including possible code sequences from the conditional instruction stream and responsive to the

initial instruction schedule would require a longer run time than the initial instruction schedule. For at least these reasons, claim 6 is allowable over the proposed combination and the rejection should be withdrawn.

Because independent claim 6 is allowable, dependent claims 7 and 8 are also allowable. *See In re Fine, supra.* Accordingly, Applicant respectfully requests that the rejection of claims 7 and 8 be withdrawn.

4. Claims 9 and 10

For convenience of analysis, independent claim 9, as amended, is repeated below in its entirety.

> 9. A computer program for performing correctness checks, the computer program being embodied on a computer-readable medium, the computer program comprising:
>
> a first code segment configured to receive a set of instructions;
>
> ***a second code segment configured to generate an initial instruction schedule and a conditional instruction stream from the set of instructions***, such that the initial instruction schedule is devoid of code sequences comprising correctness check functions and such that the code sequences of the conditional instruction stream are associated with a corresponding set of one or more instructions in the initial instruction schedule;
>
> ***a third code segment configured to evaluate the initial instruction schedule*** to determine whether the initial instruction schedule includes spare instruction slots into which said code sequences associated with the correctness check function can be inserted into the initial instruction schedule ***such that a final instruction schedule would not require a longer run time than the initial instruction schedule***; and
>
> a fourth code segment configured to insert code sequences associated with the correctness check function into the spare instruction slots when sufficient spare instruction slots exist in the initial instruction schedule to accommodate said code sequences.

(Applicant's independent Claim 9 - *emphasis added.*)

The cited art of record fails to disclose, teach, or suggest at least the emphasized elements of pending claim 9 as shown above. Consequently, claim 9 is allowable.

Specifically, the systems disclosed in *Hooker* and *Blasciak* fail to disclose, teach, or suggest Applicant's claimed "***second code segment configured to generate an initial instruction schedule and a conditional instruction stream from the set of instructions . . .***" Both *Hooker* and *Blasciak* are silent regarding generating an initial schedule and a conditional instruction stream from the first set of instructions.

Furthermore, the systems disclosed in *Hooker* and *Blasciak* fail to disclose, teach, or suggest Applicant's claimed "***third code segment configured to evaluate the initial instruction schedule . . . such that a final instruction schedule would not require a longer run time than the initial instruction schedule***." In this regard, both *Hooker* and *Blasciak* are silent regarding evaluating an initial instruction schedule such that a final instruction schedule would not require a longer run time than the initial instruction schedule. *Hooker*, as described above, merely teaches inserting tabulation instructions in a first instruction set. *Hooker* executes the instructions in a first execution unit and processes the tabulation instructions via a second execution unit. Inserting and processing tabulation instructions, as apparently taught by *Hooker*, does not disclose, teach, or suggest Applicant's claimed third code segment configured to evaluate a first instruction set. *Blasciak*, apparently teaches the insertion of code markers and linking instructions to the markers that can be used to verify correctness of time-based margins, variable references, context switches, and branch conditions. *Blasciak* cannot disclose, teach, or suggest Applicant's claimed third code segment configured to evaluate the initial instruction schedule for at least the reason that *Blasciak* results in an intrusion into the underlying software (*i.e.*, increased execution time), whereas Applicant's claimed code segment evaluates the initial instruction schedule such that a final instruction schedule responsive to the initial instruction schedule would not require a longer run time than the initial instruction schedule. For at least these reasons, claim 9 is allowable over the proposed combination and the rejection should be withdrawn.

Because independent claim 9 is allowable, dependent claim 10 is also allowable. *See In re Fine, supra.* Accordingly, Applicant respectfully requests that the rejection of claims 9 and 10 be withdrawn.

III. New Claims 11-15

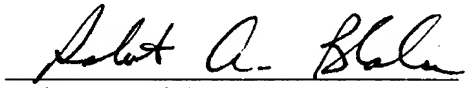
Applicant's new claims 11-14 depend directly or indirectly from allowable independent claim 1. Applicant's new claim 15 depends directly from allowable independent claim 4. Thus, new claims 11-15 are also allowable. *See In re Fine, supra.*

CONCLUSION

In summary, Applicant respectfully requests that all outstanding claim rejections be withdrawn. Applicants respectfully submit that all pending claims 1-15 are allowable over the cited art of reference and the present application is in condition for allowance. Accordingly, a Notice of Allowance is respectfully solicited. Should the Examiner have any comment regarding the Applicant's response or believe that a teleconference would expedite prosecution of the pending claims, Applicant requests that the Examiner telephone Applicant's undersigned attorney.

Respectfully submitted,

THOMAS, KAYDEN, HORSTEMEYER & RISLEY, L.L.P.

By: 

Robert A. Blaha
Registration No. 43,502
(770) 933-9500